

BOD
BOD

BOD

BOD
BOD

BOD
BOD

BOD
BOD

BOD
BOD

BOD
BOD

BOD BUG

**BUG
BYE**

**BYP
CAN**

CAN
CAN

**CAN
CAN**

**CAN
CHE**

CHE
CHE

CHE

CLU

CLU

CLU

CLU

CLU

CLU
CLU

CLU

C111

CLU
CLU

CLU
CLU

CLU
CLU

CLU
CLU

CLU
CLU

CLU
CLU

CCU

CLU

CLU
CLU

[illegible]

```

LL          IIIIII          SSSSSSSS
LL          IIIIII          SSSSSSSS
LL          III          SS
LL          III          SS
LL          III          SS
LL          III          SS
LL          III          SSSSSS
LL          III          SSSSSS
LL          III          SS
LL          III          SS
LL          III          SS
LL          III          SS
LLLLLLLLLLLL IIIIIIII SSSSSSSS
LLLLLLLLLLLL IIIIIIII SSSSSSSS

```

```
0001 0 MODULE OPC$SHARE_DEVNAME ( %TITLE 'SHARE_FULL_DEVNAME'
0002 0 , LANGUAGE (BLISS32),
0003 0 IDENT = 'V04-000'
0004 0 ) =
0005 0
0006 0 *****
0007 0 *
0008 0 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0009 0 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0010 0 * ALL RIGHTS RESERVED.
0011 0 *
0012 0 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0013 0 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0014 0 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0015 0 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0016 0 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0017 0 * TRANSFERRED.
0018 0 *
0019 0 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0020 0 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0021 0 * CORPORATION.
0022 0 *
0023 0 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0024 0 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0025 0 *
0026 0 *
0027 0 *****
0028 0
0029 0 ++
0030 0 FACILITY:
0031 0
0032 0 OPCOM
0033 0
0034 0 ABSTRACT:
0035 0
0036 0 This file contains routines shared by OPCOM, REPLY and REQUEST.
0037 0 Each routine is compiled as a separate module so that unnecessary
0038 0 code is not included at link time.
0039 0
0040 0 Environment:
0041 0
0042 0 VAX/VMS operating system.
0043 0
0044 0 Author:
0045 0
0046 0 CW Hobbs
0047 0
0048 0 Creation date:
0049 0
0050 0 31 July 1983
0051 0
0052 0 Revision history:
0053 0
0054 0 V03-003 CWH3169 CW Hobbs 5-May-1984
0055 0 Second pass for cluster-wide OPCOM:
0056 0 - Make DVIS_xxxNAM an input item for SHARE_FULL_DEVNAME so
0057 0 that calling routine can pick a name.
```



```

58      0058 0 | - Delete the entire timeout count module, it is no longer
59      0059 0 | necessary with the queued $brkthru mechanism.
60      0060 0 |
61      0061 0 | V03-002 CWH3002 CW Hobbs 16-Sep-1983
62      0062 0 | Add timeout routine and VM jacket routines.
63      0063 0 |
64      0064 0 | --
65      0065 1 BEGIN
66      0066 1
67      0067 1 LIBRARY 'SYS$LIBRARY:LIB';
68      0068 1 LIBRARY 'LIB$OPCOMLIB';
69      0069 1
70      0070 1 GLOBAL ROUTINE SHARE_FULL_DEVNAME (DEVNAME : $ref_bblock, NAME_CODE) = %SBTTL 'share_full_devname (devname,
71      0071 1
72      0072 1 !++
73      0073 1 Functional description:
74      0074 1
75      0075 1 Take the device name passed as input, and return a fully qualified device spec. This will include
76      0076 1 the SCS nodename if that is defined. Note that the descriptor is OWN storage inside of this routine
77      0077 1 so that this routine is non-reentrant.
78      0078 1
79      0079 1 Input:
80      0080 1
81      0081 1 DEVNAME : Address of a quadword buffer descriptor that
82      0082 1 points to the buffer containing the device name
83      0083 1 NAME_CODE : DVI$_xxxNAM, the device name item code
84      0084 1
85      0085 1 Implicit Input:
86      0086 1
87      0087 1 None.
88      0088 1
89      0089 1 Output:
90      0090 1
91      0091 1 None.
92      0092 1
93      0093 1 Implicit output:
94      0094 1
95      0095 1 None.
96      0096 1
97      0097 1 Side effects:
98      0098 1
99      0099 1 None.
100     0100 1
101     0101 1 Routine value:
102     0102 1
103     0103 1 Descriptor pointing to fullname. If unable to modify name, input descriptor will returned.
104     0104 1 --
105     0105 1
106     0106 2 BEGIN ! Start of SHARE_FULL_DEVNAME
107     0107 2
108     0108 2 OWN
109     0109 2 NAME_BUFFER : VECTOR [MAX_DEV_NAM, BYTE]
110     0110 2 NAME_DESC : VECTOR [2, LONG] INITIAL (0, NAME_BUFFER),
111     0111 2 SEC_NAME_BUFFER : VECTOR [MAX_DEV_NAM, BYTE]
112     0112 2 SEC_NAME_DESC : VECTOR [2, LONG] INITIAL (0, SEC_NAME_BUFFER),
113     0113 2 SPOOLED : LONG,
114     0114 2 DVI_LIST : VECTOR [10, LONG] INITIAL (

```

```

115 0115 2 MAX_DEV_NAM, ! Add item code at run time
116 0116 2 NAME_BUFFER,
117 0117 2 NAME_DESC,
118 0118 2 MAX_DEV_NAM, ! Add code at run time
119 0119 2 SEC_NAME_BUFFER,
120 0120 2 SEC_NAME_DESC,
121 0121 2 ((DVI$ SPL OR DVI$C_SECONDARY)^16 OR 4), ! Is the secondary spooled?
122 0122 2 SPOOLED,
123 0123 2 0,
124 0124 2 0);
125 0125 2
126 0126 2
127 0127 2 ! Get the name, if any problem then return the address of the input descriptor
128 0128 2
129 0129 2 (DVI_LIST [0])<16,16,0> = .NAME CODE; ! Put name item code in high word
130 0130 2 (DVI_LIST [3])<16,16,0> = (.NAME CODE OR DVI$C_SECONDARY);
131 0131 2 IF NOT ($GETDVIW (DEVNAM=.DEVNAME, ITMLST=DVI_LIST))
132 0132 2 THEN
133 0133 2 RETURN .DEVNAME;
134 0134 2
135 0135 2 IF .NAME_DESC [0] EQL 0
136 0136 2 THEN
137 0137 2 RETURN .DEVNAME;
138 0138 2
139 0139 2 ! If the device is spooled, return secondary characteristics
140 0140 2
141 0141 2 IF .SPOOLED
142 0142 2 THEN
143 0143 2 IF .SEC_NAME_DESC [0] NEQ 0
144 0144 2 THEN
145 0145 2 RETURN SEC_NAME_DESC;
146 0146 2
147 0147 2 RETURN NAME_DESC; ! If we get this far, it's ok.
148 0148 1 END; ! End of SHARE_FULL_DEVNAME

```

```

.TITLE OPC$SHARE_DEVNAME SHARE_FULL_DEVNAME
.IDENT \V04-000\
.PSECT $OWNS,NOEXE,2

```

```

00000 NAME_BUFFER:
00000000 00040 NAME_DESC: .BLKB 64
00000000' 00044 .LONG 0
00048 SEC_NAME_BUFFER: .ADDRESS NAME_BUFFER
00000000 00088 SEC_NAME_DESC: .BLKB 64
00000000' 0008C .LONG 0
00090 SPOOLED: .ADDRESS SEC_NAME_BUFFER
00000040 00094 DVI_LIST: .BLKB 4
00000000' 00000000' 00098 .LONG 64
00000000' 00000040 000A0 .ADDRESS NAME_BUFFER, NAME_DESC
00000000' 00000000' 000A4 .LONG 64
00000000' 00000000' 000A4 .ADDRESS SEC_NAME_BUFFER, SEC_NAME_DESC

```

			00530004	000AC		.LONG	5439492		
			00000000	000B0		.ADDRESS	SPOOLED		
			00000000	000B4		.LONG	0, 0		
						.EXTRN	SYSS\$GETDVIW		
						.PSECT	\$CODE\$,NOWRT,2		
						.ENTRY	SHARE_FULL_DEVNAME, Save R2		0070
						MOVAB	DVI_LIST, R2		
						MOVW	NAME_CODE, DVI_LIST+2		0129
						BISW3	#1, NAME_CODE, DVI_LIST+14		0130
						CLRQ	-(SP)		0131
						CLRQ	-(SP)		
						PUSHL	R2		
						PUSHL	DEVNAME		
						CLRQ	-(SP)		
						CALLS	#8, SYSS\$GETDVIW		
						BLBC	R0, 1\$		
						TSTL	NAME_DESC		0135
						BNEQ	2\$		
						MOVL	DEVNAME, R0		0137
						RET			
						BLBC	SPOOLED, 3\$		0141
						TSTL	SEC_NAME_DESC		0143
						BEQL	3\$		
						MOVAB	SEC_NAME_DESC, R0		0145
						RET			
						MOVAB	NAME_DESC, R0		0147
						RET			0148

; Routine Size: 68 bytes, Routine Base: \$CODE\$ + 0000

; 149 0149 1 END
; 150 0150 0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	188	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	68	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	Symbols		Pages Mapped	Processing Time
	Total	Loaded Percent		

VAX-11 Bliss-32 V4.0-742 Page 5
DISK\$VMSMASTER:[OPCOM.SRC]SHARE.B32;1 (1)

```

:                                     COMMAND QUALIFIERS
:      BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:SHARE/OBJ=OBJ$:SHARE MSRC$:SHARE/UPDATE=(ENH$:SHARE)

```

OPC\$SHARE_FAOBU SHARE_FAO_BUFFER
V03-001 share_fao_buffer (ctrstr, args)

N 6
16-Sep-1984 01:53:46
14-Sep-1984 12:50:55

VAX-11 Bliss-32 V4.0-742 Page 6
DISK\$VMSMASTER:[OPCOM.SRC]SHARE.B32;1 (2)

```

152 0151 0 MODULE OPC$SHARE_FAOBUF (IDENT = 'V03-001') = %TITLE 'SHARE_FAO_BUFFER' %SBTTL 'share_fao_buffer (ct
153 0152 1 BEGIN
154 0153 1
155 0154 1 LIBRARY 'SYS$LIBRARY:LIB';
156 0155 1 LIBRARY 'LIB$OPCOMLIB';
157 0156 1
158 0157 1 GLOBAL ROUTINE SHARE_FAO_BUFFER (ctrstr : REF VECTOR[2], args : VECTOR [4]) = %SBTTL 'SHARE_FAO_BUFFER'
159 0158 2 BEGIN
160 0159 2 ++
161 0160 2
162 0161 2 FUNCTIONAL DESCRIPTION:
163 0162 2
164 0163 2 This routine passes an ascii string through the FAO system service with any number of specified para
165 0164 2
166 0165 2 INPUTS:
167 0166 2
168 0167 2 ctrstr Address of FAO control string descriptor
169 0168 2 args Any number of additional arguments
170 0169 2
171 0170 2 IMPLICIT INPUTS:
172 0171 2
173 0172 2 none
174 0173 2
175 0174 2 OUTPUTS:
176 0175 2
177 0176 2 none
178 0177 2
179 0178 2 IMPLICIT OUTPUTS:
180 0179 2
181 0180 2 none
182 0181 2
183 0182 2 ROUTINE VALUE:
184 0183 2
185 0184 2 Address of formatted descriptor
186 0185 2
187 0186 2 SIDE EFFECTS:
188 0187 2
189 0188 2 none
190 0189 2 --
191 0190 2 OWN
192 0191 2 DESC : VECTOR [2, LONG],
193 0192 2 FAOBUF : VECTOR [512, BYTE]
194 0193 2 ;
195 0194 2
196 0195 2 DESC [0] = 512; ! Set up result descriptor
197 0196 2 DESC [1] = FAOBUF;
198 0197 2
199 0198 2 $FAOL (CTRSTR=.CTRSTR, OUTLEN=DESC, OUTBUF=DESC, PRMLST=ARGS);
200 0199 2
201 0200 2 RETURN DESC;
202 0201 1 END;
```

.TITLE OPC\$SHARE_FAOBUF SHARE_FAO_BUFFER
.IDENT \V03-001\
.PSECT \$OWNS,NOEXE,2

0201

COMMAND QUALIFIERS

OPC\$SHARE_FAOBU SHARE_FAO_BUFFER
V03-001 SHARE_FAO_BUFFER

C 7
16-Sep-1984 01:53:46
14-Sep-1984 12:50:55

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[OPCOM.SRC]SHARE.B32;1 Page 8
(2)

; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LISS:SHARE/OBJ=OBJ\$:SHARE MSRC\$:SHARE/UPDATE=(ENH\$:SHARE)

OP
V0

OPC\$SHARE_VM
V03-001

OPC\$SHARE_VM
OPC\$GET_VM (len, addr)

D 7
16-Sep-1984 01:53:46
14-Sep-1984 12:50:55

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[OPCOM.SRC]SHARE.B32;1
Page 9
(3)

OP
V0

```

207 0205 0 MODULE OPC$SHARE_VM (IDENT = 'V03-001') = %TITLE 'OPC$SHARE_VM' %SBTTL 'OPC$GET_VM (len, addr)'
208 0206 1 BEGIN
209 0207 1
210 0208 1 LIBRARY 'SYSS$LIBRARY:LIB';
211 0209 1 LIBRARY 'LIB$:OPCOMLIB';
212 0210 1
213 0211 1 GLOBAL ROUTINE OPC$GET_VM (len, addr) =
214 0212 2 BEGIN
215 0213 2 ++
216 0214 2
217 0215 2 FUNCTIONAL DESCRIPTION:
218 0216 2 This routine calls LIB$GET_VM
219 0217 2
220 0218 2 --
221 0219 2 LOCAL
222 0220 2 STATUS;
223 0221 2
224 0222 2 STATUS = LIB$GET_VM (.LEN, .ADDR);
225 0223 2
226 0224 2 RETURN .STATUS;
227 0225 2
228 0226 1 END;
```

.TITLE OPC\$SHARE_VM OPC\$SHARE_VM
.IDENT \V03-001\

.EXTRN OPC\$GET_VM, LIB\$GET_VM

.PSECT \$CODE\$,NOWRT,2

.ENTRY OPC\$GET_VM, Save nothing
MOVQ LEN, -(SP)
CALLS #2, LIB\$GET_VM
RET

: 0211
: 0223
: 0226

```

00000000G 7E 04 AC 7D 00002
02 FB 00006
04 0000D
```

; Routine Size: 14 bytes, Routine Base: \$CODE\$ + 0000

```

229 0227 1 GLOBAL ROUTINE OPC$FREE_VM (len, addr) = %SBTTL 'OPC$FREE_VM (len, addr)'
230 0228 1 BEGIN
231 0229 2
232 0230 2 ++
233 0231 2
234 0232 2 FUNCTIONAL DESCRIPTION:
235 0233 2 This routine calls LIB$FREE_VM
236 0234 2
237 0235 2 --
238 0236 2 LOCAL
239 0237 2 STATUS;
240 0238 2
241 0239 2 STATUS = LIB$FREE_VM (.LEN, .ADDR);
242 0240 2
243 0241 2 RETURN .STATUS;
244 0242 2
245 0243 1 END;
```


OPC\$SHARE_VM
V03-001

OPC\$SHARE_VM
OPC\$FREE_VM (len, addr)

E 7
16-Sep-1984 01:53:46
14-Sep-1984 12:50:55

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[OPCOM.SRC]SHARE.B32;1

Page 10
(3)

.EXTRN OPC\$FREE_VM, LIB\$FREE_VM

.ENTRY OPC\$FREE_VM, Save nothing
MOVQ LEN, -(SP)
CALLS #2, LIB\$FREE_VM
RET

: 0228
: 0240
: 0243

; Routine Size: 14 bytes, Routine Base: \$CODE\$ + 000E

: 246 0244 1
: 247 0245 1 END
: 248 0246 0 ELUDOM

! End of OPC\$SHARE_VM

PSECT SUMMARY

Name	Bytes	Attributes
\$CODE\$	28	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	Total	Symbols Loaded	Percent	Pages Mapped	Processing Time
\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	0	0	1000	00:01.8
\$255\$DUA28:[OPCOM.OBJ]OPCOMLIB.L32;1	633	4	0	43	00:00.7

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD, INITIAL, OPTIMIZE)/LIS=LIS\$:SHARE/OBJ=OBJ\$:SHARE MSRC\$:SHARE/UPDATE=(ENH\$:SHARE)

```

250 0247 0 MODULE OPC$SHARE_LOOKOPERBIT (IDENT = 'V03-001') = %TITLE 'SHARE_LOOKOPERBIT' %SBTTL 'share_init_o
251 0248 1 BEGIN
252 0249 1
253 0250 1 LIBRARY 'SYSS$LIBRARY:LIB';
254 0251 1 LIBRARY 'LIB$:OPCOMLIB';
255 0252 1
256 0253 1 EXTERNAL ROUTINE
257 0254 1 share_trnlog : NOVALUE;
258 0255 1
259 0256 1 LITERAL
260 0257 1 MAX_OPERBIT = 24;
261 0258 1
262 0259 1 OWN
263 0260 1 OPER_NAME : VECTOR [MAX_OPERBIT, LONG] PRESET (
264 0261 1 [0] = OPC$_CENTRL, [1] = OPC$_PRINT, [2] = OPC$_TAPES,
265 0262 1 [3] = OPC$_DISKS, [4] = OPC$_DEVICE, [5] = OPC$_CARDS,
266 0263 1 [6] = OPC$_NETWORK, [7] = OPC$_CLUSTER, [8] = OPC$_SECURITY,
267 0264 1 [9] = OPC$_REPLY, [10] = OPC$_SOFTWARE, [11] = OPC$_FILL_11,
268 0265 1 [12] = OPC$_OPER1, [13] = OPC$_OPER2, [14] = OPC$_OPER3,
269 0266 1 [15] = OPC$_OPER4, [16] = OPC$_OPER5, [17] = OPC$_OPER6,
270 0267 1 [18] = OPC$_OPER7, [19] = OPC$_OPER8, [20] = OPC$_OPER9,
271 0268 1 [21] = OPC$_OPER10, [22] = OPC$_OPER11, [23] = OPC$_OPER12);
272 0269 1
273 0270 1 GLOBAL
274 0271 1 OPER_KEYTBL : VECTOR (((MAX_OPERBIT+2)+1), LONG) PRESET ([0] = MAX_OPERBIT+2);
275 0272 1
276 0273 1 ROUTINE SHARE_INIT_OPER_KEYTBL : NOVALUE =
277 0274 2 BEGIN
278 0275 2 !++
279 0276 2
280 0277 2 FUNCTIONAL DESCRIPTION:
281 0278 2
282 0279 2 This routine performs run-time initializations on the operator name keyword table.
283 0280 2
284 0281 2 INPUTS:
285 0282 2
286 0283 2 None.
287 0284 2
288 0285 2 IMPLICIT INPUTS:
289 0286 2
290 0287 2 none
291 0288 2
292 0289 2 OUTPUTS:
293 0290 2
294 0291 2 none
295 0292 2
296 0293 2 IMPLICIT OUTPUTS:
297 0294 2
298 0295 2 none
299 0296 2
300 0297 2 ROUTINE VALUE:
301 0298 2
302 0299 2 none
303 0300 2
304 0301 2 SIDE EFFECTS:
305 0302 2
306 0303 2 none

```

```

307      0304      !--
308      0305
309      0306      LOCAL
310      0307      buff : VECTOR [64, BYTE],
311      0308      desc : VECTOR [2, LONG] PRESET ([1] = buff),
312      0309      msg,
313      0310      len,
314      0311      adr : REF VECTOR [, BYTE],
315      0312      status;
316      0313
317      0314      Loop through the name vector, for each non-zero entry fetch the message text and create a string
318      0315      from the message. Store a pointer to the string descriptor back in the vector.
319      0316
320      0317      INCR i FROM 0 TO MAX_OPERBIT-1
321      0318      DO
322      0319      BEGIN
323      0320      IF (msg = .oper_name [i]) EQL 0      ! We assume no holes
324      0321      THEN
325      0322      $signal_stop (ss$_badparam);
326      0323
327      0324      Reset the buffer length, and get the message text. Any problem is fatal.
328      0325
329      0326      desc [0] = 64;
330      0327      IF NOT (status = $getmsg (msgid=.msg, msglen=desc, bufadr=desc, flags=1))
331      0328      THEN
332      0329      $signal_stop (.status);
333      0330
334      0331      Allocate and initialize a counted string, store info in the keytbl
335      0332
336      0333      len = .desc [0] + 1;      ! String plus ASCII count byte
337      0334      IF NOT (status = OPC$GET_VM (len, adr))      ! Get memory, store address right into keytable
338      0335      THEN
339      0336      $signal_stop (.status);
340      0337      adr [0] = .desc [0];      ! Set the ascii count byte
341      0338      CH$MOVE (.desc [0], .desc [1], adr [1]);      ! Copy the string to the new memory
342      0339      oper_keytbl [(2*i)+1] = .adr;      ! Store the address in the keytbl
343      0340      oper_keytbl [(2*i)+2] = .i;      ! Store the index in the keytbl
344      0341      END;
345      0342
346      0343      RETURN;
347      0344      END;

```

.TITLE OPC\$SHARE_LOOKOPERBIT SHARE_LOOKOPERBIT
.IDENT \V03-001\

.PSECT \$OWNS\$,NOEXE,2

00058123	0005811B	00058113	0005810B	00058103	000580FB	00000	OPER_NAME:	
00058153	0005814B	00058143	0005813B	00058133	0005812B	00018	.LONG	360699, 360707, 360715, 360723, 360731, -
00058183	0005817B	00058173	0005816B	00058163	0005815B	00030		360739, 360747, 360755, 360763, 360771, -
000581B3	000581AB	000581A3	0005819B	00058193	0005818B	00048		360779, 360787, 360795, 360803, 360811, -
								360819, 360827, 360835, 360843, 360851, -
								360859, 360867, 360875, 360883

.PSECT \$GLOBAL\$,NOEXE,2


```
00000030 00000 OPER_KEYTBL::
                                .LONG 48
                                00004 .BLKB 192
                                .EXTRN SHARE TRNLOG, LIB$STOP
                                .EXTRN SYS$GETMSG, OPC$GET_VM
                                .PSECT $CODE$,NOWRT,2
```

```
03FC 00000 SHARE_INIT_OPER_KEYTBL:
                                .WORD Save R2,R3,R4,R5,R6,R7,R8,R9
                                MOVAB -80(SP), SP
                                CLRL DESC
                                MOVAB BUFF, DESC+4
                                CLRL I
                                59 0000'CF46 DO 00010 1$: MOVL OPER_NAME[I], MSG
                                04 12 00016 BNEQ 2$
                                14 DD 00018 PUSHL #20
                                35 11 0001A BRB 4$
                                08 AE 40 8F 9A 0001C 2$: MOVZBL #64, DESC
                                7E 01 7D 00021 MOVQ #1, -(SP)
                                10 AE 9F 00024 PUSHAB DESC
                                14 AE 9F 00027 PUSHAB DESC
                                59 DD 0002A PUSHL MSG
                                00 05 FB 0002C CALLS #5, SYS$GETMSG
                                58 DO 00033 MOVL R0, STATUS
                                16 58 E9 00036 BLBC STATUS, 3$
                                04 AE 08 AE 01 C1 00039 ADDL3 #1, DESC, LEN
                                5E DD 0003F PUSHL SP
                                08 AE 9F 00041 PUSHAB LEN
                                02 FB 00044 CALLS #2, OPC$GET_VM
                                50 DO 00049 MOVL R0, STATUS
                                58 E8 0004C BLBS STATUS, 5$
                                58 DD 0004F 3$: PUSHL STATUS
                                00 01 FB 00051 4$: CALLS #1, LIB$STOP
                                57 04 00058 RET
                                6E DO 00059 5$: MOVL ADR, R7
                                01 A7 0C BE 08 AE 90 0005C MOVAB DESC, (R7)
                                50 50 00060 MOVAB3 DESC, @DESC+4, 1(R7)
                                56 0000'CF40 01 78 00067 ASHL #1, I, R0
                                50 0000'CF40 57 DO 0006B MOVL R7, OPER_KEYTBL+4[R0]
                                91 56 00071 ASHL #1, I, R0
                                56 56 00075 MOVL I, OPER_KEYTBL+8[R0]
                                17 F3 0007B AOBLEQ #23, I, -1$
                                04 0007F RET
```

; Routine Size: 128 bytes, Routine Base: \$CODE\$ + 0000

```

349 0345 1 GLOBAL ROUTINE SHARE_LOOKUP_OPER_BIT (TEXT : $ref_bblock) = %SBTTL 'share_lookup_oper_bit'
350 0346 BEGIN
351 0347 ++
352 0348
353 0349 FUNCTIONAL DESCRIPTION:
354 0350
355 0351 This routine converts a text string for an operator name into that operators bit index in
356 0352 the operator attention mask.
357 0353
358 0354 INPUTS:
359 0355
360 0356 None.
361 0357
362 0358 IMPLICIT INPUTS:
363 0359
364 0360 none
365 0361
366 0362 OUTPUTS:
367 0363
368 0364 none
369 0365
370 0366 IMPLICIT OUTPUTS:
371 0367
372 0368 none
373 0369
374 0370 ROUTINE VALUE:
375 0371
376 0372 none
377 0373
378 0374 SIDE EFFECTS:
379 0375
380 0376 none
381 0377 --
382 0378
383 0379 LOCAL
384 0380 idx,
385 0381 status;
386 0382
387 0383 IF .oper_keytbl [1] EQL 0
388 0384 THEN
389 0385 share_init_oper_keytbl (); ! Initialize the oper_keytbl structure
390 0386
391 0387 Translate the name if possible
392 0388
393 0389 share_trnlog (.text);
394 0390
395 0391 Convert the name to the index value stored in the keyword table
396 0392
397 0393 status = lib$lookup_key (.text, oper_keytbl, idx); ! Use the library routine
398 0394 IF NOT .status
399 0395 THEN
400 0396 $signal_stop (opc$_valuerr, 1, .text, .status, 1, .text);
401 0397
402 0398 RETURN .idx;
403 0399 END;

```

			0000	00000		.EXTRN	LIB\$LOOKUP_KEY	
	SE		04	C2	00002	.ENTRY	SHARE_LOOKUP_OPER_BIT, Save nothing	0345
		0000'	CF	D5	00005	SUBL2	#4, SP	
			05	12	00009	TSTL	OPER_KEYTBL+4	0383
FF70	CF		00	FB	0000B	BNEQ	1\$	
		04	AC	DD	00010	CALLS	#0, SHARE_INIT_OPER_KEYTBL	0385
0000G	CF		01	FB	00013	PUSHL	TEXT	0389
			5E	DD	00018	CALLS	#1, SHARE_TRNLOG	
		0000'	CF	9F	0001A	PUSHL	SP	0393
		04	AC	DD	0001E	PUSHAB	OPER_KEYTBL	
00000000G	00		03	FB	00021	PUSHL	TEXT	
	1A		50	EB	00028	CALLS	#3, LIB\$LOOKUP_KEY	
		04	AC	DD	0002B	BLBS	STATUS, 2\$	0394
			01	DD	0002E	PUSHL	TEXT	0396
		04	50	DD	00030	PUSHL	#1	
			AC	DD	00032	PUSHL	STATUS	
			01	DD	00035	PUSHL	TEXT	
00000000G	00	0005825C	BF	DD	00037	PUSHL	#1	
			06	FB	0003D	PUSHL	#3\$1052	
				04	00044	CALLS	#6, LIB\$STOP	
	50		6E	DC	00045	RET	IDX, R0	0398
			04	00048	RET			0399

; Routine Size: 73 bytes, Routine Base: \$CODE\$ + 0080

: 404 0400 1 END
: 405 0401 0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	96	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$GLOBALS	196	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	201	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	Total	Symbols Loaded	Percent	Pages Mapped	Processing Time
\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	30	0	1000	00:01.8
\$255\$DUA28:[OPCOM.OBJ]OPCOMLIB.L32;1	633	4	0	43	00:00.7

OPC\$SHARE_LOOKO SHARE_LOOKOPERBIT
V03-001 share_lookup_oper_bit

K 7
16-Sep-1984 01:53:46
14-Sep-1984 12:50:55

VAX-11 BLISS-32 V4.0-742
DISK\$VMSMASTER:[OPCOM.SRC]SHARE.B32;1 Page 16
(5)

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:SHARE/OBJ=OBJ\$:SHARE MSRC\$:SHARE/UPDATE=(ENH\$:SHARE)

OP
VO

```

407 0402 0 MODULE OPC$SHARE_INITOPERNAME (IDENT = 'V03-001') = %TITLE 'SHARE_INIT_OPER_NAME' %SBTTL 'share_init_o
408 0403 1 BEGIN
409 0404 1
410 0405 1 LIBRARY 'SYSS$LIBRARY:LIB';
411 0406 1 LIBRARY 'LIB$OPCOMLIB';
412 0407 1
413 0408 1 GLOBAL ROUTINE SHARE_INIT_OPER_NAME : NOVALUE =
414 0409 2 BEGIN
415 0410 2 ++
416 0411 2
417 0412 2 FUNCTIONAL DESCRIPTION:
418 0413 2
419 0414 2 This routine performs run-time initializations on the operator name vector.
420 0415 2
421 0416 2 INPUTS:
422 0417 2
423 0418 2 None.
424 0419 2
425 0420 2 IMPLICIT INPUTS:
426 0421 2
427 0422 2 none
428 0423 2
429 0424 2 OUTPUTS:
430 0425 2
431 0426 2 none
432 0427 2
433 0428 2 IMPLICIT OUTPUTS:
434 0429 2
435 0430 2 none
436 0431 2
437 0432 2 ROUTINE VALUE:
438 0433 2
439 0434 2 none
440 0435 2
441 0436 2 SIDE EFFECTS:
442 0437 2
443 0438 2 none
444 0439 2 --
445 0440 2
446 0441 2
447 0442 2 Define the vector of message codes that describe the text associated with each of the known
448 0443 2 attention bits. Undefined bits have a zero message code associated with them. The order of the
449 0444 2 entries must coincide with the order of the bits that are defined, including any undefined bits.
450 0445 2 The run-time initialization of this vector is to take each non-zero entry, fetch the message text
451 0446 2 associated with that entry, and store a pointer to a string descriptor for the message text.
452 0447 2
453 0448 2 GLOBAL
454 0449 2 OPER_NAME : VECTOR [64, LONG] PRESET (
455 0450 2 [0] = OPC$_CENTRL, [1] = OPC$_PRINT, [2] = OPC$_TAPES,
456 0451 2 [3] = OPC$_DISKS, [4] = OPC$_DEVICE, [5] = OPC$_CARDS,
457 0452 2 [6] = OPC$_NETWORK, [7] = OPC$_CLUSTER, [8] = OPC$_SECURITY,
458 0453 2 [9] = OPC$_REPLY, [10] = OPC$_SOFTWARE, [11] = OPC$_FILL_11,
459 0454 2 [12] = OPC$_OPER1, [13] = OPC$_OPER2, [14] = OPC$_OPER3,
460 0455 2 [15] = OPC$_OPER4, [16] = OPC$_OPER5, [17] = OPC$_OPER6,
461 0456 2 [18] = OPC$_OPER7, [19] = OPC$_OPER8, [20] = OPC$_OPER9,
462 0457 2 [21] = OPC$_OPER10, [22] = OPC$_OPER11, [23] = OPC$_OPER12);
463 0458 2

```

```

464 0459 2 LOCAL
465 0460      buff : VECTOR [64, BYTE],
466 0461      desc : VECTOR [2, LONG] PRESET ([1] = buff),
467 0462      msg,
468 0463      adr : REF VECTOR [, LONG],
469 0464      status;
470 0465
471 0466      Loop through the name vector, for each non-zero entry fetch the message text and create a string
472 0467      from the message. Store a pointer to the string descriptor back in the vector.
473 0468
474 0469      INCR i FROM 0 TO 63
475 0470      DO
476 0471          IF (msg = .oper_name [i]) NEQ 0
477 0472          THEN
478 0473              BEGIN
479 0474                  Reset the buffer length, and get the message text. Any problem is fatal.
480 0475
481 0476                  desc [0] = 64;
482 0477                  IF NOT (status = $getmsg (msgid=msg, msglen=desc, bufadr=desc, flags=1))
483 0478                  THEN
484 0479                      $signal_stop (.status);
485 0480
486 0481                  Allocate and initialize a descriptor and string
487 0482
488 0483                  IF NOT (status = OPC$GET_VM (%REF (8), adr))
489 0484                  THEN
490 0485                      $signal_stop (.status);
491 0486                      oper_name [i] = .adr;
492 0487                      adr [0] = .desc [0];
493 0488                      IF NOT (status = OPC$GET_VM (adr [0], adr [1]))
494 0489                      THEN
495 0490                          $signal_stop (.status);
496 0491                          CH$MOVE (.desc [0], .desc [1], .adr [1]);
497 0492
498 0493                  END;
499 0494
500 0495      RETURN;
501 0496 1 END;

```

.TITLE OPC\$SHARE_INITOPERNAM SHARE_INIT_OPER_NAME
.IDENT \V03-001\

.PSECT \$GLOBAL\$,NOEXE,2

00058123	0005811B	00058113	0005810B	00058103	000580FB	00000	OPER_NAME::												
00058153	0005814B	00058143	0005813B	00058133	0005812B	00018		.LONG	360699,	360707,	360715,	360723,	360731,	-					
00058183	0005817B	00058173	0005816B	00058163	0005815B	00030			360739,	360747,	360755,	360763,	360771,	-					
000581B3	000581AB	000581A3	0005819B	00058193	0005818B	00048			360779,	360787,	360795,	360803,	360811,	-					
									360819,	360827,	360835,	360843,	360851,	-					
						00060			360859,	360867,	360875,	360883,							
								.BLKB	160										

.EXTRN SYS\$GETMSG, LIB\$STOP
.EXTRN OPC\$GET_VM

.PSECT \$CODE\$,NOWRT,2

				01FC 00000	.ENTRY	SHARE_INIT_OPER_NAME, Save R2,R3,R4,R5,R6,-	0408
	5E	B0	AE	9E 00002	MOVAB	R7,R8	
		08	AE	D4 00006	CLRL	-80(SP), SP	
0C	AE	10	AE	9E 00009	CLRL	DESC	0461
			56	D4 0000E	MOVAB	BUFF, DESC+4	
	58	0000'CF	46	D0 00010	CLRL	I	0469
			64	13 00016	MOVL	OPER_NAME[I], MSG	0471
			8F	9A 00018	BEQL	4\$	
08	AE	40	01	7D 0001D	MOVZBL	#64, DESC	0477
	7E		01	7D 0001D	MOVQ	#1, -(SP)	0478
		10	AE	9F 00020	PUSHAB	DESC	
		14	AE	9F 00023	PUSHAB	DESC	
			58	DD 00026	PUSHL	MSG	
00000000G	00		05	FB 00028	CALLS	#5, SYSSGETMSG	
	57		50	D0 0002F	MOVL	R0, STATUS	
	36		57	E9 00032	BLBC	STATUS, 2\$	
		04	AE	9F 00035	PUSHAB	ADR	0484
04	AE		08	D0 00038	MOVL	#8, 4(SP)	
		04	AE	9F 0003C	PUSHAB	4(SP)	
0000G	CF		02	FB 0003F	CALLS	#2, OPC\$GET_VM	
	57		50	D0 00044	MOVL	R0, STATUS	
	21		57	E9 00047	BLBC	STATUS, 2\$	
0000'CF	46	04	AE	D0 0004A	MOVL	ADR, OPER_NAME[I]	0487
04	BE	08	AE	D0 00051	MOVL	DESC, @ADR	0488
	52	04	AE	D0 00056	MOVL	ADR, R2	0489
		04	A2	9F 0005A	PUSHAB	4(R2)	
		08	AE	DD 0005D	PUSHL	ADR	
0000G	CF		02	FB 00060	CALLS	#2, OPC\$GET_VM	
	57		50	D0 00065	MOVL	R0, STATUS	
	0A		57	E8 00068	BLBS	STATUS, 3\$	
			57	DD 0006B	PUSHL	STATUS	0491
00000000G	00		01	FB 0006D	CALLS	#1, LIB\$STOP	
			04	00074	RET		
04	B2	0C	BE	08	AE	28 00075	3\$: 0492
	90		56	3F	F3 0007C	4\$: 0471	
				04	00080	RET	0496

; Routine Size: 129 bytes. Routine Base: \$CODE\$ + 0000

: 502 0497 1 END
: 503 0498 0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
\$GLOBALS	256	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	129	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
..\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	27	0	1000	00:01.8
..\$255\$DUA28:[OPCOM.OBJ]OPCOMLIB.L32;1	633	2	0	43	00:00.8

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:SHARE/OBJ=OBJ\$:SHARE MSRC\$:SHARE/UPDATE=(ENH\$:SHARE)

```

0499 0 MODULE OPC$SHARE_TRNLOG (IDENT = 'V03-001') = %TITLE 'SHARE_TRNLOG' %SBTTL 'share_trnlog (name)'
0500 1 BEGIN
0501 1
0502 1 LIBRARY 'SYSS$LIBRARY:LIB';
0503 1 LIBRARY 'LIB$:OPCOMLIB';
0504 1
0505 1 GLOBAL ROUTINE SHARE_TRNLOG (NAME : $ref_bblock) : NOVALUE =
0506 1 ++
0507 1 Functional description:
0508 1
0509 1 Recursively translate a logical name.
0510 1
0511 1 Input:
0512 1
0513 1 name - Address of dynamic string descriptor for input name
0514 1
0515 1 Implicit Input:
0516 1
0517 1 None.
0518 1
0519 1 Output:
0520 1
0521 1 name - Receives a new dynamic string
0522 1
0523 1 Implicit output:
0524 1
0525 1 None.
0526 1
0527 1 Side effects:
0528 1
0529 1 None.
0530 1
0531 1 Routine value:
0532 1
0533 1 None.
0534 1 --
0535 1
0536 2 BEGIN ! Start of share_trnlog
0537 2
0538 2 LOCAL
0539 2 lcl_buf : $bvector [256],
0540 2 in_dsc : VECTOR [2, LONG],
0541 2 out_dsc : VECTOR [2, LONG],
0542 2 status;
0543 2
0544 2 If the input string was is not dynamic, scream and shout.
0545 2
0546 2 IF .name [dsc$b_class] NEQ dsc$b_k_class_d
0547 2 THEN
0548 2 $signal_stop (ss$badparam);
0549 2
0550 2 Copy the input string to the local buffer
0551 2
0552 2 CH$MOVE (.name [dsc$b_length], .name [dsc$a_pointer], lcl_buf);
0553 2
0554 2 Set the input and output descriptors up
0555 2

```

```

562 0556 2 in_dsc [0] = .name [dsc$w_length];
563 0557 2 in_dsc [1] = lcl_buf;
564 0558 2 out_dsc [1] = lcl_buf;
565 0559 2
566 0560 2 Try up to ten times to translate the name
567 0561 2
568 0562 2 INCR i FROM 0 TO 10
569 0563 2 DO
570 0564 2 BEGIN
571 0565 2
572 0566 2 We didn't find the end, give an error
573 0567 2
574 0568 2 IF .i GEQ 10
575 0569 2 THEN
576 0570 2 $signal_stop (opc$_valuerr, 1, .name, ss$_toomanylnam);
577 0571 2
578 0572 2 Set up the output descriptor
579 0573 2
580 0574 2 out_dsc [0] = 256;
581 0575 2
582 0576 2 Attempt to translate
583 0577 2
584 0578 2 status = $strnlog (lognam=in_dsc, rslten=out_dsc, rslbuf=out_dsc);
585 0579 2 IF .status EQL ss$_notran ! No translation, we are done
586 0580 2 THEN
587 0581 2 EXITLOOP;
588 0582 2 IF NOT .status
589 0583 2 THEN
590 0584 2 $signal_stop (.status);
591 0585 2
592 0586 2 Get ready for the next loop
593 0587 2
594 0588 2 in_dsc [0] = .out_dsc [0];
595 0589 2 END;
596 0590 2
597 0591 2 Copy the local string to the output
598 0592 2
599 0593 2 IF NOT (status = str$copy_dx (.name, out_dsc))
600 0594 2 THEN
601 0595 2 $signal_stop (.status);
602 0596 2
603 0597 2 RETURN;
604 0598 1 END;

```

.TITLE OPC\$SHARE_TRNLOG SHARE_TRNLOG
.IDENT \V03-001\

.EXTRN LIB\$STOP, SYS\$STRNLOG
.EXTRN STR\$COPY_DX

.PSECT \$CODE\$,NOWRT,2

.ENTRY SHARE_TRNLOG, Save R2,R3,R4,R5,R6,R7 : 0505
MOVAB LIB\$STOP, R7 :
MOVAB -272(SP), SP :
MOVL NAME, R6 : 0546

57 00000000G 00 9E 00002
5E FEFO CE 9E 00009
56 04 AC DO 0000E

		02	03	A6	91	00012		CMPB	3(R6), #2		
				04	13	00016		BEQL	1\$		
				14	DD	00018		PUSHL	#20		0548
				6B	11	0001A		BRB	6\$		
10	AE	04	B6	66	28	0001C	1\$:	MOVCB	(R6), @4(R6), LCL_BUF		0552
		08	AE	66	3C	00022		MOVZWL	(R6), IN_DSC		0556
		0C	AE	10	AE	9E	00026	MOVAB	LCL_BUF, IN_DSC+4		0557
		04	AE	10	AE	9E	0002B	MOVAB	LCL_BUF, OUT_DSC+4		0558
				52	D4	00030		CLRL	I		0562
		0A		52	D1	00032	2\$:	CMPL	I, #10		0568
				13	19	00035		BLSS	3\$		
		7E	0374	8F	3C	00037		MOVZWL	#884, -(SP)		0570
				56	DD	0003C		PUSHL	R6		
				01	DD	0003E		PUSHL	#1		
			0005825C	8F	DD	00040		PUSHL	#361052		
		67		04	FB	00046		CALLS	#4, LIB\$STOP		
				04	00049			RET			
		6E	0100	8F	3C	0004A	3\$:	MOVZWL	#256, OUT_DSC		0574
				7E	7C	0004F		CLRQ	-(SP)		0578
				7E	D4	00051		CLRL	-(SP)		
			0C	AE	9F	00053		PUSHAB	OUT_DSC		
			10	AE	9F	00056		PUSHAB	OUT_DSC		
			1C	AE	9F	00059		PUSHAB	IN_DSC		
	00000000G	00		06	FB	0005C		CALLS	#6, SYS\$TRNLOG		
	00000629	8F		50	D1	00063		CMPL	STATUS, #1577		0579
				0B	13	0006A		BEQL	4\$		
		16		50	E9	0006C		BLBC	STATUS, 5\$		0582
	08	AE		6E	D0	0006F		MOVL	OUT_DSC, IN_DSC		0588
BB		52		0A	F3	00073		AOBLEQ	#10, I, 2\$		0562
			4040	8F	BB	00077	4\$:	PUSHR	#*M<R6,SP>		0593
	00000000G	00		02	FB	0007B		CALLS	#2, STR\$COPY_DX		
		05		50	E8	00082		BLBS	STATUS, 7\$		
				50	DD	00085	5\$:	PUSHL	STATUS		0595
		67		01	FB	00087	6\$:	CALLS	#1, LIB\$STOP		
				04	0008A	7\$:		RET			0598

; Routine Size: 139 bytes, Routine Base: \$CODE\$ + 0000

; 605 0599 1 END
; 606 0600 0 ELUDOM ! End of SHARE_TRNLOG

PSECT SUMMARY

Name	Bytes	Attributes
\$CODE\$	139	NOVEC,NOWRT, RD, EXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2)

Library Statistics

OPC\$SHARE_TRNLO SHARE_TRNLOG
V03-001 share_trnlog (name)

F 8
16-Sep-1984 01:53:46
14-Sep-1984 12:50:55

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[OPCOM.SRC]SHARE.B32;1 Page 24
(7)

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	12	0	1000	00:01.8
-\$255\$DUA28:[OPCOM.OBJ]OPCOMLIB.L32;1	633	4	0	43	00:00.7

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:SHARE/OBJ=OBJ\$:SHARE MSRC\$:SHARE/UPDATE=(ENH\$:SHARE)

: Size: 603 code + 1256 data bytes
: Run Time: 00:32.2
: Elapsed Time: 02:11.0
: Lines/CPU Min: 1116
: Lexemes/CPU-Min: 12055
: Memory Used: 101 pages
: Compilation Complete

0292 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY